Linux Network Servers

Firewall

Nos tempos atuais tem se falado muito em segurança, pois a internet se tornou um ambiente perigoso. Todos nossos servidores que estão expostos para a internet necessitam de uma proteção para que não exponha os serviços que estão ali rodando e muito menos informações importantes sobre a empresa. A configuração de um firewall depende diretamente da disponibilidade de serviços de rede e roteamento.

Criar um estrutura de configuração para um firewall nem sempre é uma tarefa simples. Se você ainda não tem um conhecimento básico sólido em Redes é necessário estudar mais para que não ocorra maiores dificuldades na implementação do mesmo. Para configurar um firewall, é necessário o conhecimento sobre a estrutura da rede em questão e dos diferentes protocolos envolvidos na comunicação, isto é, dos serviços que a rede usa para que eles não percam a comunicação.

O objetivo em ter uma máquina fazendo o papel de Firewall Gateway em nossa é rede é minimizar as tentativas de ataques que nossas redes recebem, tentando impedir possíveis invasões e levantamento de informações. Os sistemas GNU/Linux com Kernel série 2.4 e 2.6 trabalham com o Iptables para fazer o gerenciamente de regras de Firewall. Lembrando que o Iptables é apenas um Front-End que gerencia o suporte Netfilter no Kernel. Um firewall faz o filtro de pacotes que passam na rede.

Características do iptables:

- **Filtro de pacotes statefull:** isso significa que o iptables é capaz de atuar sobre as camadas do protocolo TCP;
- **Modularidade:** a configuração do kernel é modular e com o netfilter não é diferente, pois novas funcionalidades podem ser adicionadas em muito esforço. Um módulo só será usado se for da necessidade do administrador;
- O Iptables possui as seguintes tabelas, sendo elas: **filter, nat, mangle.** A tabela **filter** é a tabela padrão do Iptables. Cada uma dessas tabelas possui o que chamamos de **CHAINS**. As CHAINS são onde vão ser definidos as regras para o nosso firewall.

A tabela filter serve para atribuir permissões de acessos essenciais (permitir/negar). A tabela NAT, que significa Network Address Translation, é um recurso que permite compartilhar acessos de Internet ou redirecionar conexões. Já a table mangle é utilizada para modificar uma propriedade de um pacote e seu uso é avançado, como por exemplo influenciar na decisão de roteamento ou controle de banda.

Linux Network Servers

As CHAINS da tabela filter são as seguintes:

INPUT	Regras de entrada de pacotes.
OUTPUT	Regras de saída de pacotes.
FORWARD	Regras de passagem de pacotes pelo firewall.

As CHAINS da tabela nat são as seguintes:

PREROUTING	Regras que serão processadas antes do roteamento dos pacotes nas interfaces do firewall.
POSTROUTING	Regras que serão precessadas pós roteamento dos pacotes nas interfaces do firewall.
OUTPUT	Regras de saída de pacotes.

Fluxo de verificações em que um pacote é submetido

Quando um pacote chega ao firewall, a primeira chain verificada é a PREROUTING. É exatamente nesse momento que algumas decisões de roteamento podem acontecer, exemplo: um redirecionamento de conexão ou de porta. Dependendo do destino, o pacote pode ser verificado na chain INPUT ou FORWARD. A chain INPUT é usada quando o destino é o próprio firewall, senão é usada a chain FORWARD que é um encaminhamento (roteamento). Se a chain INPUT é executada, o próximo passo é que chain OUTPUT seja processada, pois aí é que vai a resposta. A chain POSTROUTING é a última a ser processada, que é o momento antes de o pacote ser entregue ao destino.

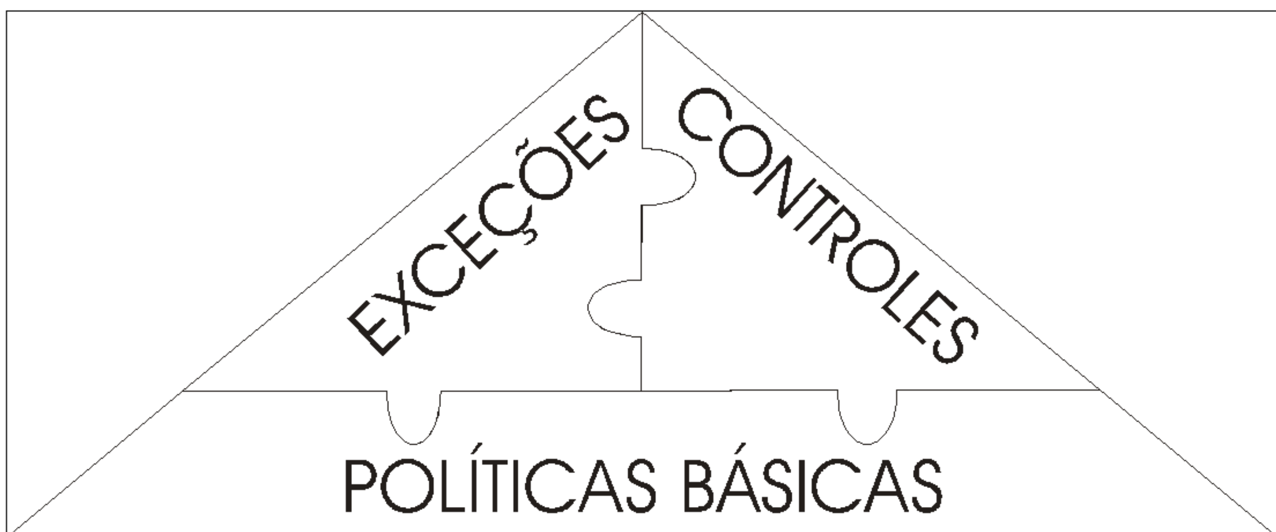
Importante: A chain PREROUTING é a primeira a ser analisada e a POSTROUTING a última. Não é possivel utilizar as chains PREROUTING e POSTROUTING na tabela filter. Na tabela nat o redirecionamento de conexões é feita na chain PREROUTING e para compartilhar acesso usa-se a chain POSTROUTING.

Linux Network Servers

Compreendendo as políticas BÁSICAS e o conceito das EXCEÇÕES

A metodologia utilizada para implementação do firewall será a seguinte:

Iremos negar todo o tráfego para as CHAINS de **INPUT**, **OUTPUT** e **FORWARD** da tabela filter, posteriormente iremos definir a relação dos serviços que devem ser liberados no firewall, a estes, iremos chamar de exceções. Todo o tráfego de pacotes que as nossas exceções não cobrir serão bloqueado por padrão. Em suma, o que não for oficialmente permitido já está expressamente negado.



Linux Network Servers

Sintaxe do comando iptables:

```
# iptables [-t tabela] [opção] [chain] [dados] -j [alvo]
```

Parâmetros para o iptables		Descrição do parâmetro
-P	--policy	Estabelece a política de acesso de uma chain
-t	--table	Seleciona tabela
-A	--append	Adiciona como última regra da sequência de uma chain
-I	--insert	Insere como primeira regra da sequência de uma chain
-N	--new-chain	Cria uma nova chain
-D	--delete	Remove uma regra
-X	--delete-chain	Elimina todas as regras presentes em chains de usuário
-F	--flush	Elimina todas as regras presentes em uma chain padrão (INPUT, FORWARD etc) ou tabela (para todas as chains)
-s	--source	Determina a origem do pacote
-d	--destination	Determina o destino do pacote
--dport	--destination-port	Define a porta de destino
--sport	--source-port	Define a porta de origem
-i	--in-interface	Define a interface de entrada (input), exemplos: eth0, eth1, ppp0 etc.
-o	--out-interface	Define a interface de saída (output)
-p	--protocol	Seleciona protocolo (tcp, udp, icmp etc)

Alvos:

Alvo (target)	Descrição do alvo
ACCEPT	O pacote é aceito
REJECT	O pacote é rejeitado imediatamente
DROP	O pacote é negado silenciosamente (mais interessante, pois diminui a eficiência de um ataque DOS/DDOS, isto é, o host de origem fica sem resposta até cair por tempo esgotado.

Linux Network Servers

Exemplos:

Verifique como estão configuradas as políticas básicas que estão definidas por padrão:

```
# iptables -n -L
```

Modifique as políticas básicas para DROP ALL:

```
# iptables -P INPUT DROP
# iptables -P OUTPUT DROP
# iptables -P FORWARD DROP
```

Verifique se a nova política foi assumida:

```
# iptables -n -L
```

Agora que percebemos que temos um firewall ativo, devemos pensar nas demais políticas, uma vez que, por mais seguro que seja um firewall, cuja política base seja negar tudo, não é um firewall prático, pois precisamos realizar comunicações. Dessa forma, precisamos definir políticas de exceções para o Firewall.

Realize o teste usando o comando ping na sua interface loopback:

```
# ping 127.0.0.1
```

O teste anterior nos permitiu verificar que devemos definir uma política de exceção para a interface loopback. Criaremos uma política que possibilite isso:

```
# iptables -A OUTPUT -d 127.0.0.1 -j ACCEPT
# iptables -A INPUT -d 127.0.0.1 -j ACCEPT
```

Liste as políticas ativas:

```
# iptables -n -L
```

Liste as políticas ativas:

```
# iptables -n -L
```

Vejamos se agora conseguimos fazer um ping na intreface de loopback:

```
# ping 127.0.0.1
```

Linux Network Servers

Execute o comando ping, tendo como alvo o endereço uma máquina da sua rede para verificar se alguma comunicação é possível:

```
# ping 192.168.200.254
```

Agora criaremos uma política que permita que seja executado o comando ping a partir de sua máquina com a sua interface de rede interna, mas sua máquina não irá responder a ping:

```
# iptables -A  OUTPUT -p icmp --icmp-type 8 -s 192.168.200.254 -d 0/0 -j ACCEPT
# iptables -A  INPUT -p icmp --icmp-type 0 -s 0/0 -d 192.168.200.254 -j ACCEPT
```

Verifique se as regras foram adicionadas:

```
# iptables -n -L
```

Podemos ver se conseguimos fazer um ping em alguma máquina da rede:

```
# ping 192.168.200.254
```

Agora que já temos um política de exceção, tente fazer um ping no domínio www.uol.com.br:

```
# ping www.uol.com.br
```

Apesar de conseguirmos usar o ping nos endereços IP's, ainda não conseguimos fazer um ping por nomes. Vamos desenvolver a regra que faça isso:

```
# iptables -A OUTPUT -p udp -s 192.168.200.254 --sport 1024:65535 -d 0/0 --dport 53 -j
ACCEPT
# iptables -A INPUT -p udp -s 0/0 --sport 53 -d 192.168.200.254 --dport 1024:65535 -j
ACCEPT
```

Verifique se as regras foram adicionadas:

```
# iptables -n -L
```

Com as regras definidas, podemos fazer um ping por nomes:

```
# ping www.uol.com.br
```

Mesmo que liberamos o nosso firewall para resolver os nomes, ainda não conseguimos acessar um servidor Web por ele, pois precisamos liberar o acesso as portas 80 e 443.

Linux Network Servers

Criaremos uma regra de exceção que permita navegação web:

```
# iptables -A OUTPUT -p tcp -s 192.168.200.100 --sport 1024:65535 -d 0/0 --dport 80  -j
ACCEPT
# iptables -A OUTPUT -p tcp -s 192.168.200.100 --sport 1024:65535 -d 0/0 --dport 443  -j
ACCEPT

# iptables -A INPUT -p tcp -s 0/0 --sport 80 -d 192.168.200.100 --dport 1024:65535  -j
ACCEPT
# iptables -A INPUT -p tcp -s 0/0 --sport 443 -d 192.168.200.100 --dport 1024:65535  -j
ACCEPT
```

Façamos um teste para ver se conseguimos traçar rotas usando a ferramenta mtr:

```
# mtr 200.176.2.11
```

O mtr utiliza respostas icmp do tipo 11, então precisamos criar uma regra liberando a entrada desse tipo de pacote:

```
# iptables -A INPUT -p icmp --icmp-type 11 -s 0/0 -j ACCEPT
```

Verifique se a regra foi adicionada:

```
# iptables -n -L
```

Execute o comando mtr para testar a regra criada:

```
# mtr 200.17.2.11
```

Firewall como Gateway de Rede

Se o nosso servidor é, por exemplo, um firewall de fronteira entre a sua rede e a internet, ou seja, um gateway de rede, devemos estabelecer uma política que faça o repasse dos pacotes de uma rede para a outra, para permitir o repasse(forward) de pacotes entre uma rede e outra.

A primeira coisa que precisamos fazer é liberar o repasse de pacotes entre as interfaces no kernel:

```
# sysctl -a | grep ip_forward
# sysctl -w net.ipv4.ip_forward=1
```

Linux Network Servers

Obs: Para deixar esse valor fixo, devemos deixar esse parâmetro dentro de /etc/sysctl.conf

```
# vi /etc/sysctl.conf
net.ipv4.ip_forward=1
```

Agora precisamos permitir no iptables que nossa rede se comunique com outras. Devemos fazer isso acrescentas regras na chain FORWARD:

```
# iptables -A FORWARD -s 192.168.200.0/24 -j ACCEPT
# iptables -A FORWARD -d 192.168.200.0/24 -j ACCEPT
```

Não podemos esquecer que a internet trabalha com IP's reservados, diferente da nossa rede. Por isso teremos que fazer a tradução do endereçamento inválido (da LAN) para o válido (da internet), através da especificação da tabela Nat, fazendo o mascaramento.

Vamos fazer com que nossa LAN seja mascarada:

```
# iptables -t nat -A POSTROUTING -o ethX -s 192.168.200.0/24 -j MASQUERADE
```

Obs: A interface ethX é a que está com o IP válido.

Verifique como estão as regras inseridas:

```
# iptables -n -L
# iptables -n -L -t nat
```

Para não perdermos essas regras, podemos salva-lás utilizando recursos do iptables, lembrando que isso não é ainda um script profissional:

```
# iptables-save > /root/firewall.back
# cat /root/firewall.back
```

Agora podemos fazer um teste e limpar todas as regras adicionadas na memória:

```
# iptables -F
# iptables -F -t nat => Limpa somente as regras da tabela nat
```

Verifique se as regras foram apagadas:

```
# iptables -n -L
# iptables -n -L -t nat
```

Linux Network Servers

Modifique as políticas básicas para ACCEPT:

```
# iptables -P INPUT ACCEPT
# iptables -P OUTPUT ACCEPT
# iptables -P FORWARD ACCEPT
```

Script de Firewall

Todas as regras que foram feitas, ficam na memória do computador, se ele for reiniciado, perderemos todas elas. Podemos utilizar o iptables-save, mas ele não fica um script profissional.

Segue aqui um script com todas as regras que foram feitas, em seguida esse script pode ser adicionado aos níveis de execução do sistema, para ser carregado sempre a máquina for ligada.

Vamos chamar nosso script de firewall.sh:

```
# cd /etc/init.d
# vi firewall.sh
#!/bin/bash
# Firewall personalizado – 4Linux

## Definição de variáveis
IPT=$(which iptables)
ET0="192.168.200.X"
NET="0/0"
PA=1024:65535
REDE="192.168.200.0/24"

## Fechando as Políticas
$IPT -P INPUT DROP
$IPT OUTPUT DROP
$IPT FORWARD DROP
## Liberando LoopBack
$IPT -A OUTPUT -d 127.0.0.1 -j ACCEPT
$IPT -A INPUT -d 127.0.0.1 -j ACCEPT

## Liberando Ping (Saída de icmp 8 e Entrada de icmp 0)
$IPT -A  OUTPUT -p icmp -icmp-type 8 -s $ET0 -d $NET -j ACCEPT
$IPT -A  INPUT -p icmp --icmp-type 0 -s $NET -d $ET0 -j ACCEPT

## Liberando resolução de nomes
$IPT -A OUTPUT -p udp -s $ET0 --sport $PA -d $NET --dport 53 -j ACCEPT
$IPT -A INPUT -p udp -s $NET --sport 53 -d $ET0 --dport $PA -j ACCEPT
```

Linux Network Servers

```
## Liberando navegação web
$IPT -A OUTPUT -p tcp -s $ET0 --sport $PA -d $NET --dport 80 -j ACCEPT
$IPT -A OUTPUT -p tcp -s $ET0 --sport $PA -d $NET --dport 443 -j ACCEPT

$IPT -A INPUT -p tcp -s $NET --sport 80 -d $ET0 --dport $PA -j ACCEPT
$IPT -A INPUT -p tcp -s $NET --sport 443 -d $ET0 --dport $PA -j ACCEPT

## Liberando consultas mtr
$IPT -A INPUT -p icmp --icmp-type 11 -s $ET0 -j ACCEPT

## Regras de FORWARD e NAT para liberar a LAN para acessar a internet.
net.ipv4.ip_forward=1


$IPT -A FORWARD -s $REDE -j ACCEPT
$IPT -A FORWARD -d $REDE -j ACCEPT
$IPT -t nat -A POSTROUTING -o ethX -s $REDE -j MASQUERADE
```

Agora podemos setar as permissões de execução para o script:

```
# chmod 755 firewal.sh
# ls -l firewall.sh
```

Para que ele seja iniciado junto com sistema quando a máquina for ligada, podemos colocar o script nos níveis de execução:

```
# update-rc.d firewall.sh defaults
# ls -l /etc/rc2.d
```